FairMQ application example in CbmRoot

N. Winckler[1], *M. Al-Turany*[1], *D. Bertini*[1], *R. Karabowicz*[1], *D. Kresan*[1], *A. Lebedev*[1], *A. Manafov*[1], *A. Rybalchenko*[1], *and F. Uhlig*[1]

[1]GSI, Darmstadt, Germany

Introduction

The FairMQ package is an asynchronous messaging layer in FairRoot framework aiming to support online/offline data analysis with high data rates[1, 2, 3]. It allows to distribute processes on different nodes (from a laptop to an entire homogenous or heterogenous system with many thousands of cores) and provides the communication layer between these processes.

In this contribution, we present a simple FairMQ application example in the context of the CbmRoot framework[4].

Device topology

The components encapsulating the tasks are called devices and derive from the common base class FairMQDevice. The devices are arranged into topologies, which describe the data flow between the different deployed processes. In this example, the used device topology is shown in figure1 and consists of the following devices:

- Two data sources (called Sampler) sending microslices
- One Merger merging microslices into timeslices
- One FileSink receiving and storing data to file

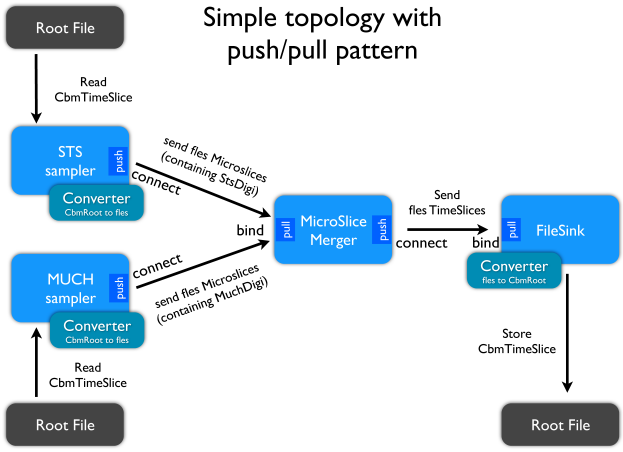

Figure 1: Simple topology involving four processes (2 samplers, 1 merger, 1 FileSink) with the push-pull messaging pattern.

Sampler, Merger and FileSink

The sampler device is used as data source, i.e. it reads data from a file and send them to another device. In this example, the two samplers read from root file the *CbmRoot timeslices*, convert the data into STS or MUCH FLES *microslices*, and then streamed them to the Merger device.

The merger device collect, synchronize and merge the FLES *microslices* streamed by the two samplers into FLES *timeslices*. Once merged, the *timeslices* are streamed to the FileSink device.

The FileSink device collect the data send by the Merger, convert them into CbmRoot data format and store the result into root file.

Summary and outlook

A simple FairMQ application example is available in the CbmRoot simulation framework. Other process nodes (e.g. track finder or track reconstruction task) as well as other messaging pattern (e.g. request/reply, pub/sub) can be easily added to the device topology. A Dynamic Deployment System (DDS) under development will ease the topology generation and deployments[5].

References

[1] D. Klein, “Flexible data transport for the online analysis in a particle physics experiment”, Bachelor Thesis, TU darmstadt (2013)

[2] A. Rybalchenko, and M. Al-Turany “Streaming data processing with FairMQ”, GSI Scientific Report 2013-2014

[3] M. Al-Turany et al. “ALFA: A new Framework for ALICE and FAIR experiments”, GSI Scientific Report 2013-2014

[4] CbmRoot collaboration

[5] A. Manafov, et al. “DDS : A Dynamic Deployment Sytem”, GSI report 2014-2015

DOI:10.15120/GR-2015-1-INFRASTRUCTURE-IT-03